

Dfs in Windows 2000

Mitch Tullock

(Reprinted from WindowsItPro Magazine)

I'm certain of three things in life: death, taxes, and file servers going belly-up for no particular reason. As much as we'd all like 100-percent reliability in our systems, today's mix of complex hardware and OSs makes this dream impossible. Microsoft has gone to great lengths to increase Windows 2000 Server's (Win2K Server's) reliability, but your servers will sometimes still crash and leave your users without access to the files they need. Dfs helps you prevent these problems.

Dfs Overview

In February 1998, I wrote about Dfs in Windows NT Server 5.0 Beta 1. (See "Get a Head Start on NT 5.0 with Dfs.") On a basic level, Dfs is a way of organizing file share resources across network servers into one logical structure that users can easily navigate. In the same way that Dfs lets you organize shares and directories on one server into a logical hierarchical structure for simplified navigation, Dfs lets you organize resources across multiple servers on your network.

You use Dfs to define a network file share that contains other file shares (as opposed to files and directories). These child (i.e., subordinate) shares and directories can be on any of your network servers—whether your systems are spread throughout your building or across the globe. When users navigate through the Dfs structure, they move from one server to another without realizing they've changed servers. Figure 1, page 102, shows a Dfs structure consisting of shares from multiple servers organized into one logical hierarchical structure. On a properly designed Dfs-enabled network, end users don't need to know which server a file is on.

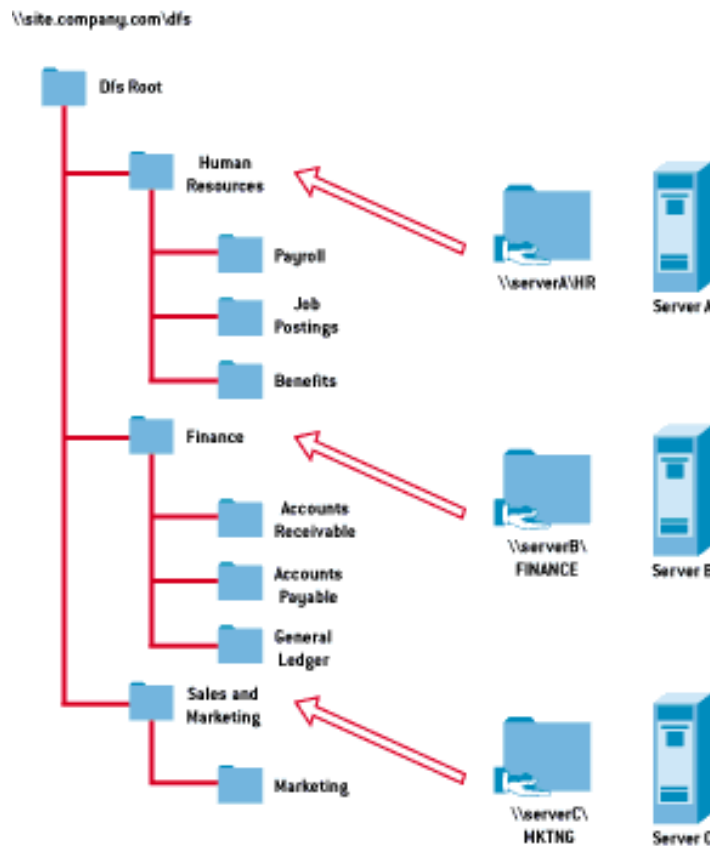


FIGURE: Toombs DFS- Winter Issue

Dfs in Windows 2000

Mitch Tullock

(Reprinted from WindowsItPro Magazine)

Because Dfs lets you tie together file resources that are spread across different systems, you can define redundant locations for those resources. For example, if you have a shared directory of documents on server A, you can place a replica of those documents on server B in a Dfs replica set. After you define a replica set, Dfs automatically redirects users navigating to that directory to a server at another site if the closest replica isn't available.

When I experimented with Dfs in 1998, the service lacked a means of replicating file changes between the locations you define in the replica set. This deficiency was important because if a user makes a change to a document on one server and Dfs doesn't replicate the change to the copies of the document on the other servers in the replica set, data corruption and versioning problems will likely occur. In Windows 2000 (Win2K) beta 3, Microsoft used a new service called the File Replication Service (FRS) to solve this problem.

Prerequisites

For Dfs to work on your network, you need at least two Win2K Server machines. You can use just one server to perform minor tests, but you need more than one system to take advantage of Dfs's organization, fault-tolerance, and load-sharing capabilities. You also need to store the data that you want to include in a replicated Dfs structure on an NTFS 5.0 volume, rather than on a FAT or regular NTFS volume. Win2K Server monitors NTFS 5.0's journal date to determine when files in a directory that the replication service needs to replicate have changed.

According to Microsoft, Win2K client systems will be able to access fault-tolerant Dfs (also referred to as domain-based Dfs) roots, in addition to standalone Dfs roots. NT 4.0 clients can access standalone Dfs roots by default and fault-tolerant Dfs roots if you install Service Pack 6 (SP6). Windows 98 clients can access standalone Dfs roots out of the box, and you can add the Active Directory (AD) client pack to upgrade these clients to access fault-tolerant roots. Win95 clients can't access any type of Dfs structure on their own. With the AD client pack installed, Win95 clients can access both types of Dfs roots.

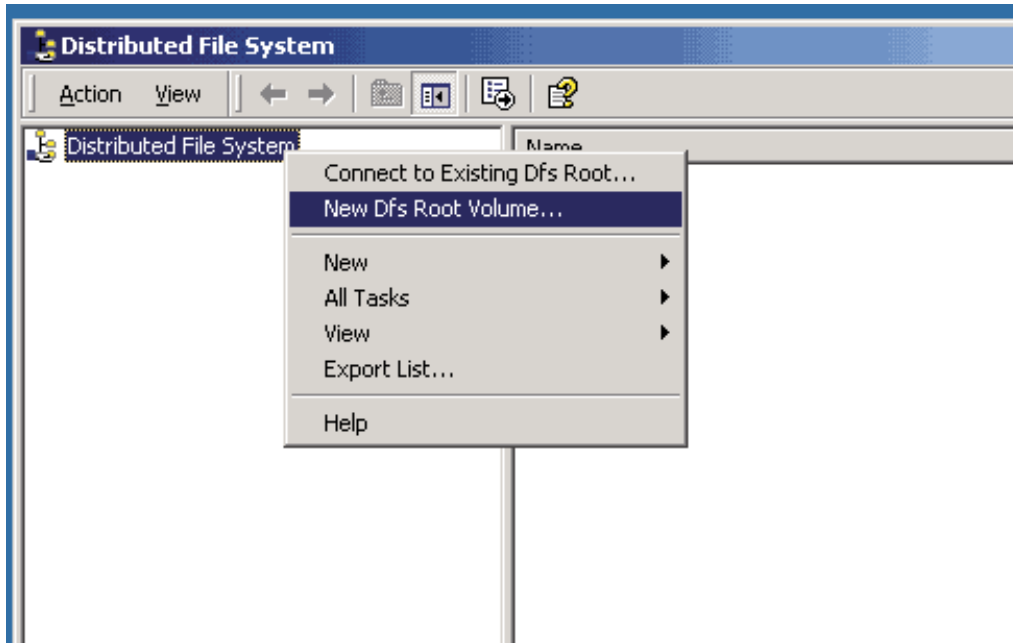
Start with the Root

All file systems and file shares have a starting point. In Dfs, the starting point is known as the Dfs root. The Dfs root is similar to the root directory on a hard disk because the Dfs root is the starting location for all other files and subdirectories. In early versions of Dfs, you could define multiple locations via replica sets to make child objects and containers redundant, but you couldn't make the root redundant. Thus, Dfs had a single-point-of-failure weakness. Microsoft revised Dfs so that you can now create redundant Dfs roots across servers. Redundant roots prevent your Dfs structure from going offline if one of your servers fails.

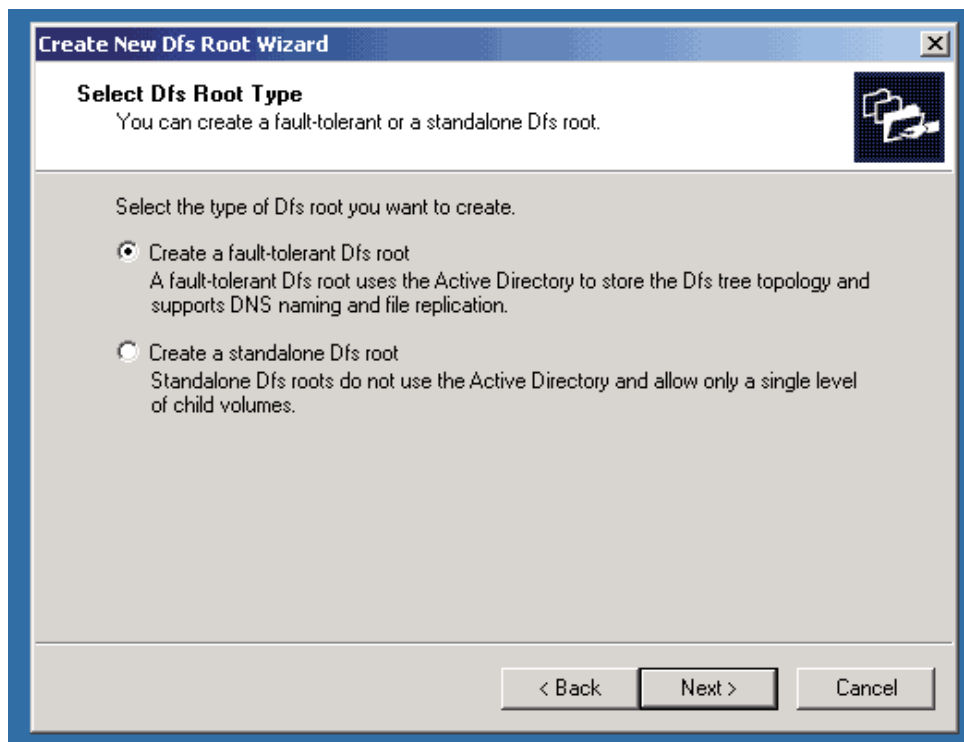
To create a redundant Dfs root, you need a working AD infrastructure. (For information about AD in Win2K, see Darren Mar-Elia, "Active Directory in Windows 2000," page 65.) Dfs uses AD's multimaster replication model to replicate data about the Dfs structure throughout the network, thus preventing the failure of one system from taking the Dfs structure offline.

Decide which servers will participate in the Dfs root, and create a share on each of these systems. After you define root shares on your servers, select Programs, Administrative Tools, Distributed File System from the Win2K Server Start menu to launch the Distributed File System window via the Microsoft Management Console (MMC). Then, select Distributed File System in the console's left pane, and choose New Dfs Root Volume from the Action menu (or right-click Distributed File System and select New Dfs Root Volume, as Screen 1 shows).

Dfs in Windows 2000
Mitch Tullock
(Reprinted from WindowsItPro Magazine)

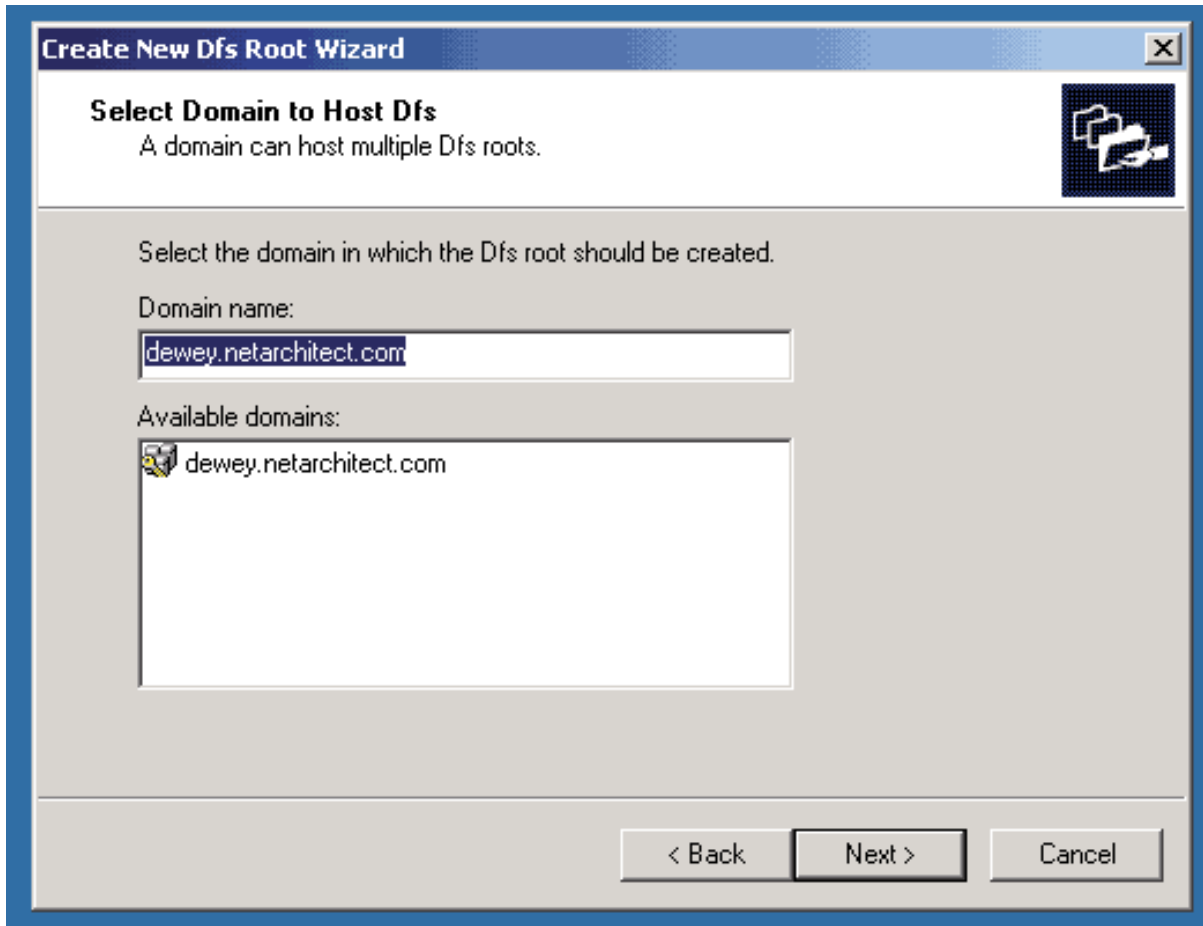


When the Create New Dfs Root Wizard window opens, you must supply the necessary information to create a Dfs root. You must first select the type of root (i.e., fault-tolerant or standalone), as Screen 2 shows. Dfs references fault-tolerant roots, which can span multiple systems, by client workstation via the domain name; Dfs references standalone roots, which exist on only one system, by the machine name hosting the Dfs root. Fault-tolerant roots are preferable, but they require a functioning AD infrastructure.



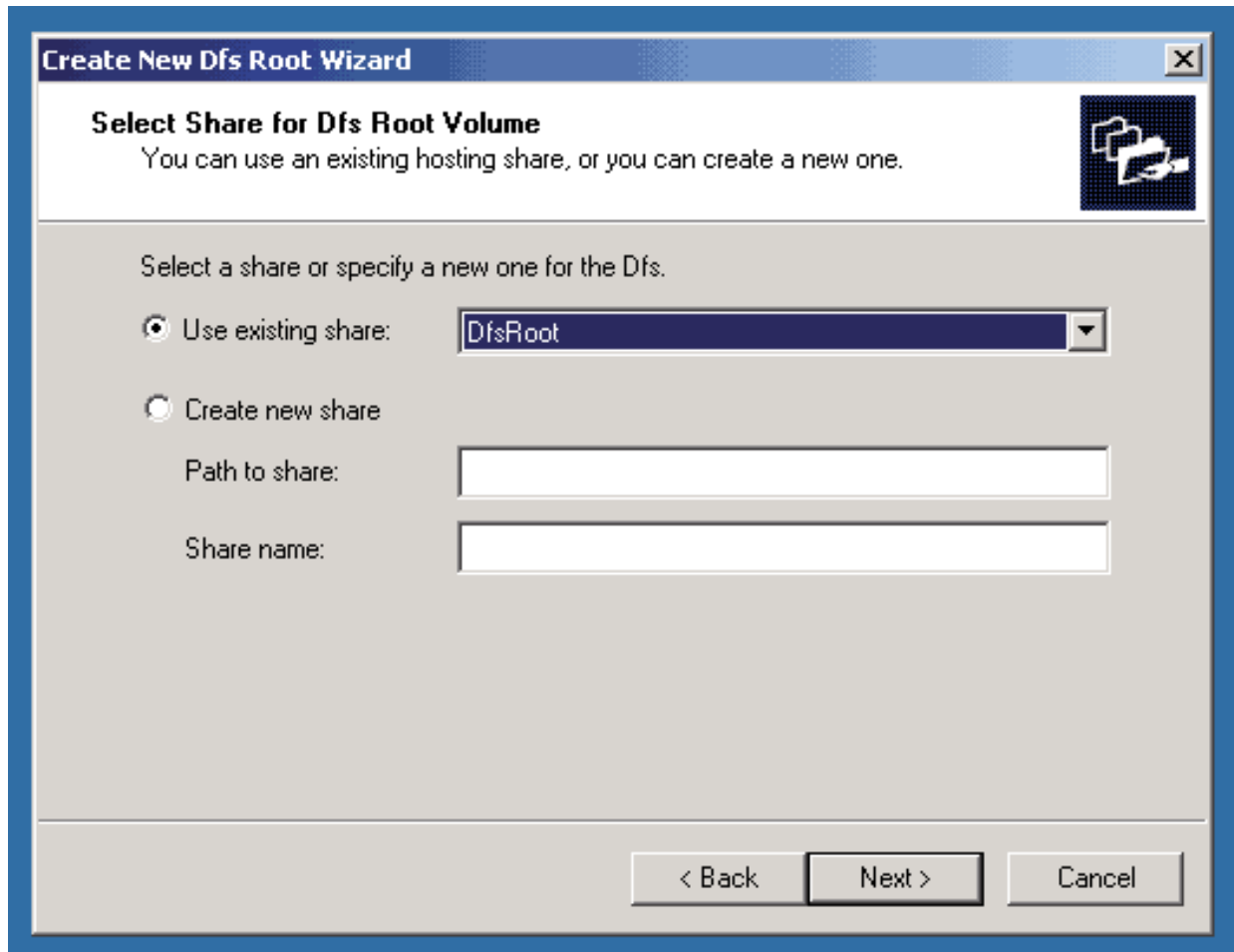
Dfs in Windows 2000
Mitch Tullock
(Reprinted from WindowsItPro Magazine)

After you specify a fault-tolerant root, the wizard prompts you for the domain to host the Dfs structure. This domain becomes the first portion of the Uniform Naming Convention (UNC) that client workstations use to connect to the file system. In the example that Screen 3 shows, I chose the dewey.netarchitect .com domain. In this example, the UNC map is \\dewey.netarchitect.com\Dfs name.



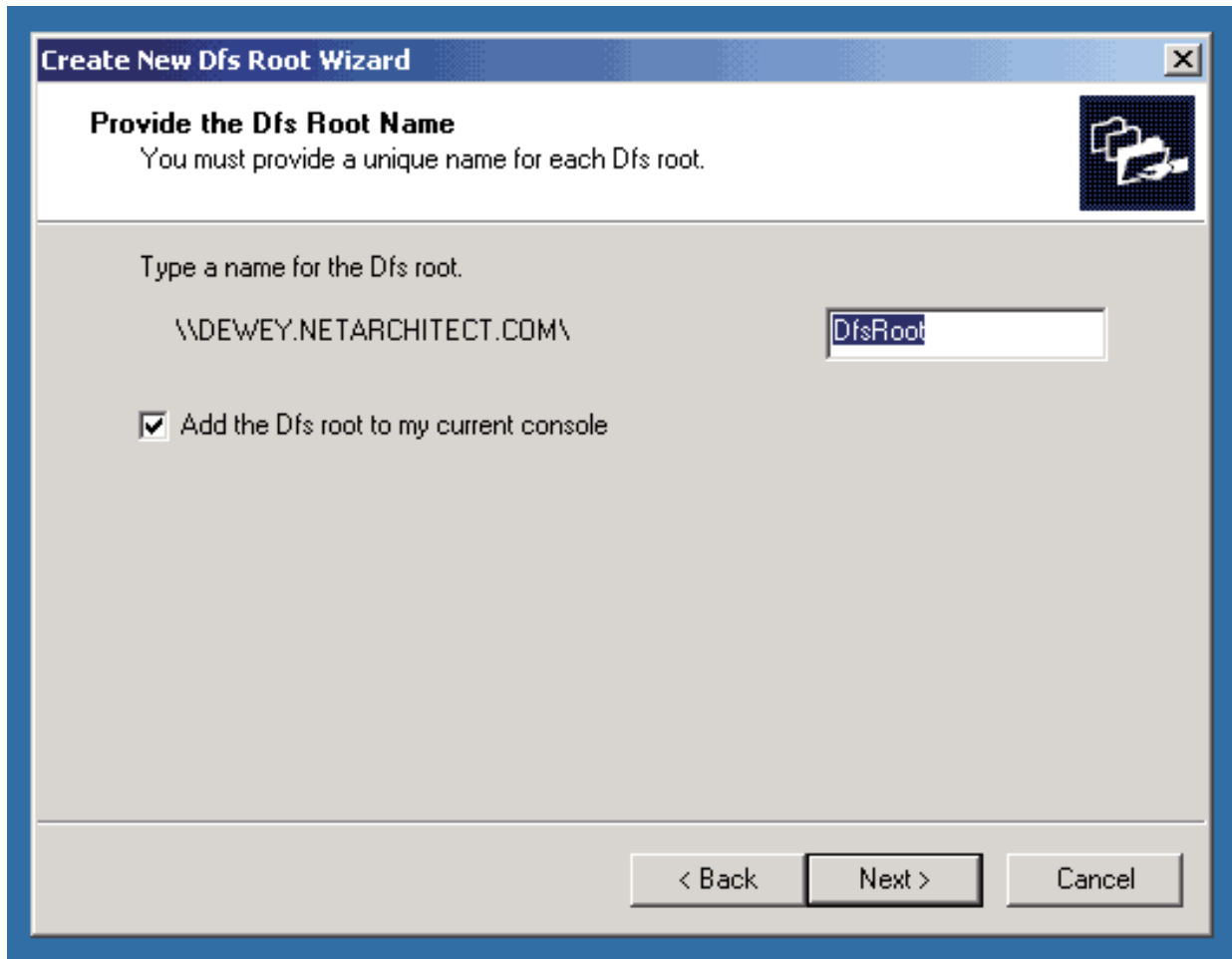
Next, you must define a server to host the Dfs root. After you specify the server to use, you need to select a share to host the root. If you followed my earlier advice, you've already created this share. (Otherwise, you can create a share by selecting the Create new share option.) Select Use existing share, and enter the name of the share you created, as Screen 4 shows.

Dfs in Windows 2000
Mitch Tullock
(Reprinted from WindowsItPro Magazine)



In the last step of the wizard, which Screen 5, page 104, shows, you define the name to assign to the Dfs root. You don't need to use the same name as the file share that defines the root, because the root can be fault tolerant across many systems that might not use the same share name. However, all Dfs root names must be unique within a domain, so you need to plan the names if you'll host multiple Dfs roots. The Dfs name becomes the second part of the UNC name that client computers use to connect to the Dfs structure. After the wizard prompts you to verify the information you've entered, the program creates the Dfs root on your system.

Dfs in Windows 2000
Mitch Tullock
(Reprinted from WindowsItPro Magazine)

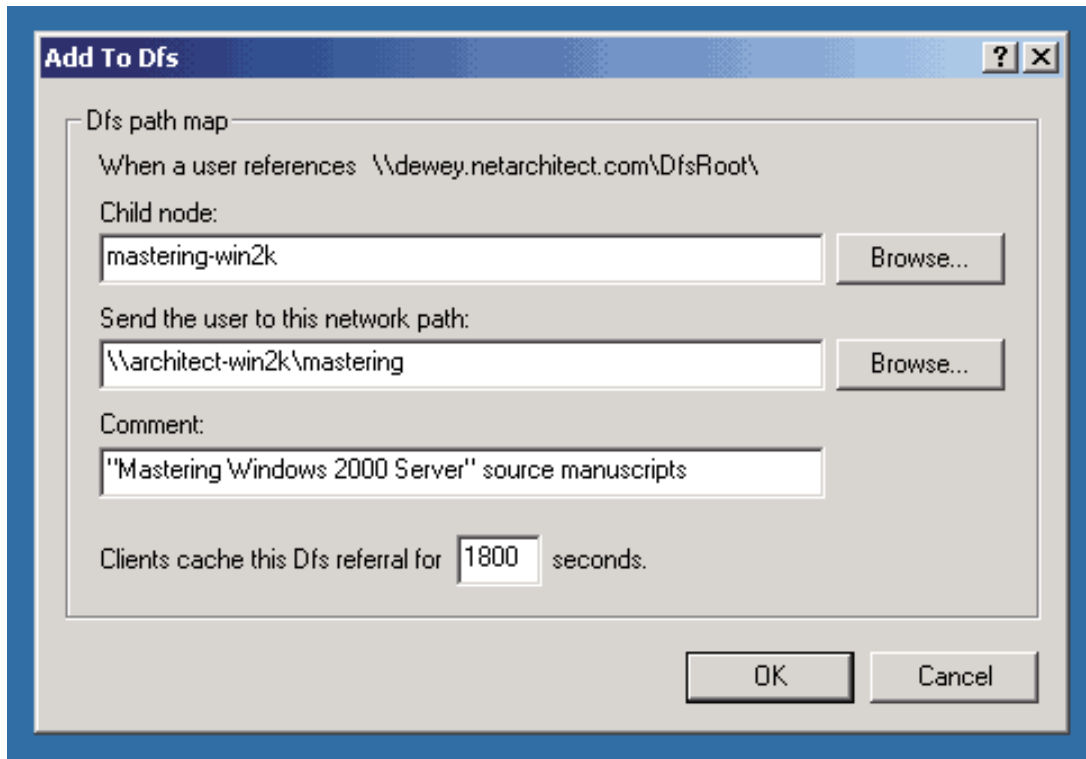


Defining the Dfs Structure

A Dfs root without child items is like a C drive without subdirectories. Thus, you need to add some structure to your Dfs system. A good approach to organizing your structure is to list all your file shares across all your servers and decide which shares you want to include in the Dfs structure and which shares you want redundant replicas of. After you compile the list, you can easily organize resources into a logical structure and define that structure within Dfs.

To define child objects in your Dfs structure, open the Distributed File System window via the MMC, select your Dfs root in the left pane, and choose New, Dfs Child Node from the Action menu. The Add To Dfs dialog box, which Screen 6, page 104, shows, opens. To define the child item, you need to provide a name for the child node (e.g., mastering-win2k). You'll want to use a descriptive name for the child node because the node will look like a directory or a file share when a user maps to the node. You must also specify the target server and file share to point users to when they select the child node. In my example in Screen 6, the target server is architect-win2k, and the file share is mastering.

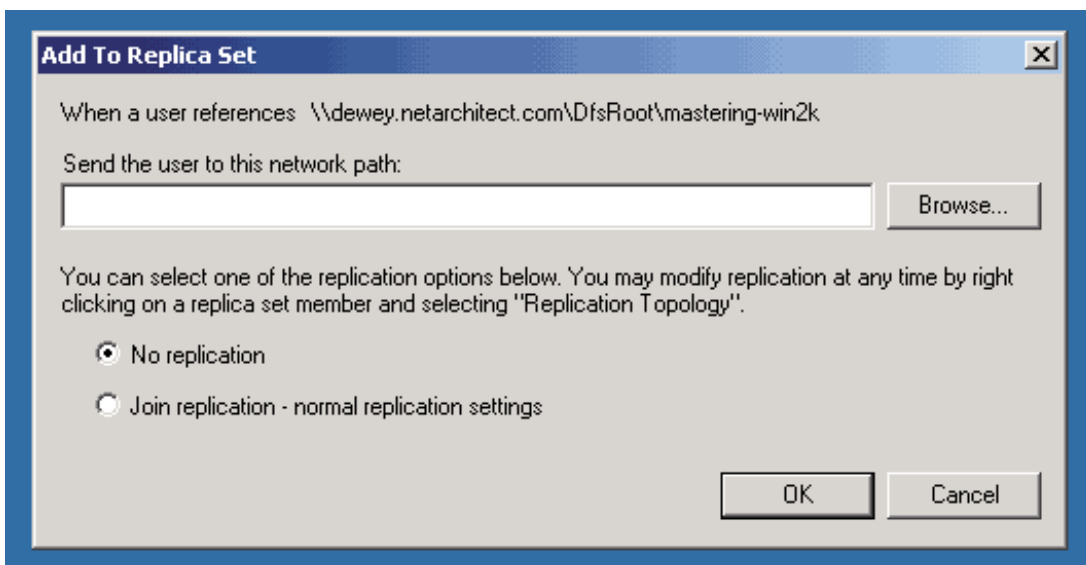
Dfs in Windows 2000
Mitch Tullock
(Reprinted from WindowsItPro Magazine)



You can add an optional comment for the Dfs child node, and you can specify a length of time for client workstations to remember the referral. The default timeout value is 1800 seconds (30 minutes), which is an acceptable length of time unless you plan to change your Dfs structure frequently.

Defining Replicas

Dfs's real power is apparent when you add an alternative location for users to find the documents they need. To make a child node redundant, open the Distributed File System window via the MMC, select the Dfs child item in the left pane, and choose New, Dfs Replica Member from the Action menu. The Add To Replica Set dialog box, which Screen 7 shows, opens.



Dfs in Windows 2000

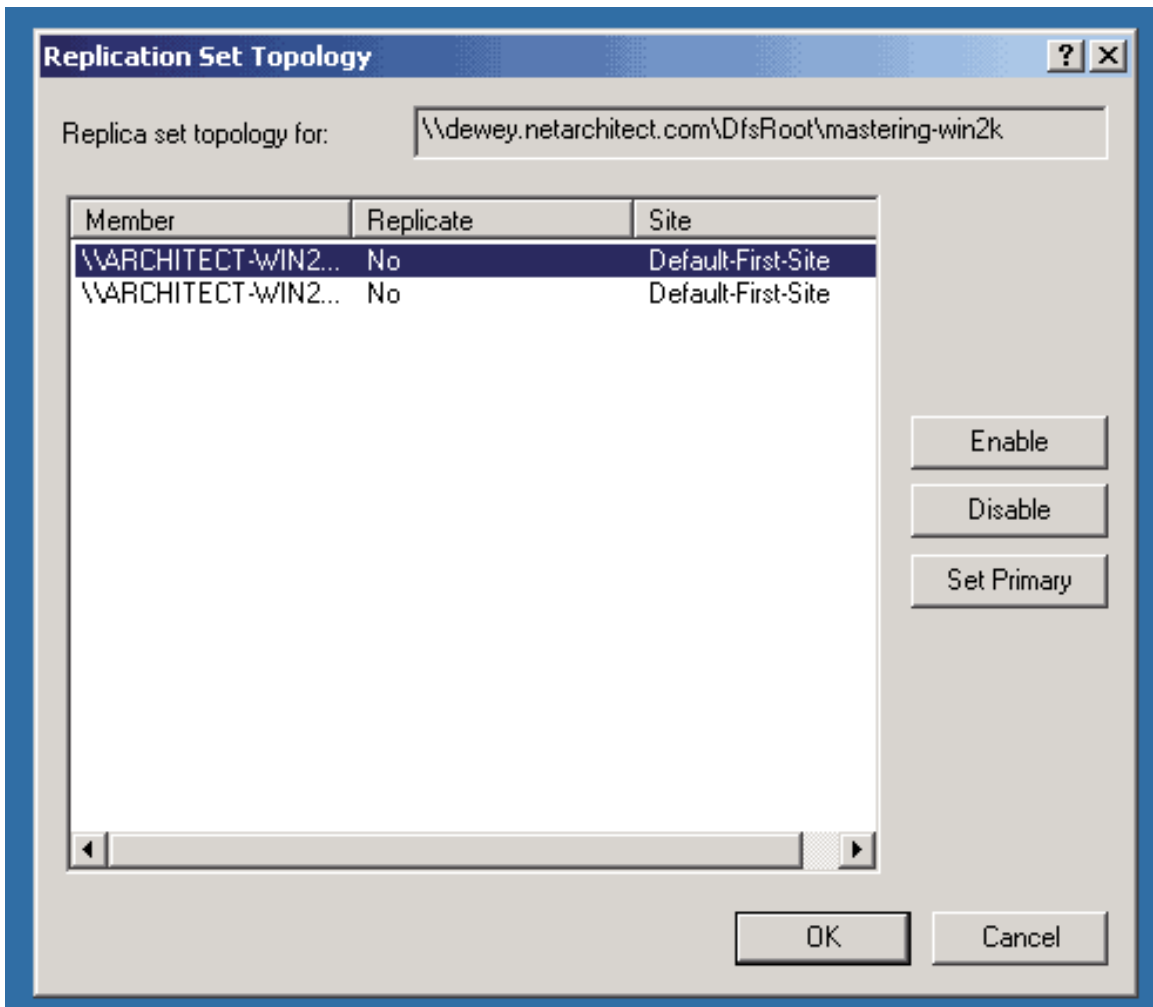
Mitch Tullock

(Reprinted from WindowsItPro Magazine)

Because the Dfs child item already has a name, comment, and timeout value, you simply need to define an alternative share location that client workstations can use to find a replica of files in the original location. Enter the share name for the alternative location, and choose a file-replication setting. If you've verified that your data is the same in both locations, I suggest selecting the Join replication option. Otherwise, select No replication. If you select No replication, you can apply the replication settings at a later time.

Tuning Replication

To fine-tune the replication settings between your systems, open the Distributed File System window via the MMC, select the Dfs child item, and choose Replication Set Topology from the Action menu. A window such as the one in Screen 8 opens, showing a list of servers and shares participating in the replica set and the settings for each item. You can enable or disable replication for each item.



Replication occurs when you change a file on a server and that server notices and propagates the change to the other systems in the replica set. Only certain changes (i.e., file, permissions, and security changes) replicate under Dfs. File and permissions changes replicate between shares in a replica set. Therefore, if a user modifies a document on one system, the modified document propagates to all the other systems participating in the Dfs replica set. Likewise, if you change

Dfs in Windows 2000

Mitch Tullock

(Reprinted from WindowsItPro Magazine)

permissions on a directory or file in a Dfs replica set, the new permissions propagate to that directory or file on all the other systems participating in the Dfs replica set.

File locks don't propagate across systems. This point is important to understand because this limitation lets two users access one file on two servers at the same time. If two users make changes to a file and save the file, one of the users' changes will be lost. File replication propagates file changes between systems in the replica set. Thus, each server propagates its changes to all the other systems, and the second user's changes overwrite the first user's changes. To prevent this problem, you need to treat alternative locations in a replica set as backup locations rather than as areas that service client requests for data.

I recommend that you not use Dfs for database-type files (e.g., Microsoft Exchange Server information stores) that you typically keep open and modify constantly. Because replication changes propagate on a file level, using Dfs for these types of files would likely wreak havoc with the file-replication service and might lead to data-corruption problems. I recommend that you use Dfs only for file-based resources that you routinely open and then close. You need to test non-client/server database files (e.g., Microsoft Access files) in your environment. Because the replication service propagates changes on a file level, your server might repropagate a 10MB database across your network every time a user modifies a file.

Accessing Redundant Dfs Resources

When multiple replicas exist for a Dfs child item, the client workstation making a request must determine which replica is closest. Dfs checks the AD for site information about the user and the share the user is trying to reach. If the first share isn't available, the client workstation continues through the list of servers until it finds a server that responds. When Dfs finds a responding server, the service presents the share to the user. This resolution process is automatic and transparent to users. When I set up a Dfs structure, I always pull a server off the network and verify that Dfs directs the requests elsewhere to test the resolution process.

Streamlining Your Job

Dfs is one of Win2K Server's most advanced services. This service makes systems administration easy because you don't have to wait for evenings or weekends, when users aren't accessing your systems, to take a server offline. Instead, you can perform system maintenance in the middle of the day because all the necessary shared data is available on other systems. With Dfs, any organization with more than one file server can easily achieve a basic level of fault tolerance, load sharing, and replication between systems.